Procedural Generation of Village Populations with Consistent and Interlinked Relationships

Austin A Flynn

BSc (Hons) Computer Games Application Development, 2021

School of Design and Informatics
Abertay University

Table of Contents

Table of Figures

Table of Tables

Acknowledgements

I would like to thank my supervisor Matthew Bett for his guidance while writing this paper.

I would also like to thank Tarn Adams for co-creating Dwarf Fortress, responding to my email, and for going so far as to recommend an excellent paper to read as research.

Most of all I would like to thank Eilidh McCreadie, who at this point has probably read this paper more than I have in an attempt to keep my sentences below 50 words, as well as being a full-time amazing girlfriend.

Abstract

As game worlds grow in size so does the demand for NPCs to fill them. Creating the hundreds of characters necessary for an open world is a time-consuming process which has the potential to divert development resources away from other areas of a game's production. Creating an application which can generate large populations of individuals that have relationships to one another, as well as the world they inhabit, could dramatically reduce the size of teams required to work on open world games. A prototype application was developed to attempt to generate a village population with a history of 100 years that would output dialogue for each individual Villager. This output was then compared with that of 2 existing games, one procedurally generated and one manually written. The testing was performed by a group of 62 participants who were not told the sources of any of the dialogue, and then ranked it on 4 separate criteria. By analysing the results of this testing, it became apparent that while the procedurally generated dialogue was ranked higher than the manually generated dialogue, creating engagement via procedurally generated dialogue proved to be difficult. In conclusion, the project was a success, however there is still significant room for improvement in both the application and data collection areas of the investigation.

Keywords: Procedural Generation, Simulation, Non-Player Characters, Dialogue, Artificial Intelligence

Abbreviations, Symbols and Notation

AI – Artificial Intelligence

MBTI – Myers-Briggs Type Indicator

NPC – Non-Player Character

RNG – Random Number Generation/Generator

RPG – Role-Playing Game

Chapter 1 - Introduction

1.1 – Project Relevance

Procedural generation is an alternative to manual content generation for games where an algorithm is followed by a game to produce nearly infinite amounts of content for a player to explore. The issue with most of this content is that due to never having direct input from a designer, it can often come off as generic or soulless, especially where personalities are involved. Non-Player Characters are an essential aspect of any open world game which demands investment in the world from a player. As worlds in games are becoming larger, the potential workload for a game designer and/or writer also increases significantly if a game wants to maintain internal consistency and a sense of shared history as opposed to having characters simply named ‘villager’. This is where procedural generation would step in. Early use of procedural generation was usually for terrain, using fractals as a basis due to their real-world parallels (Carpenter, 1980, p.109), however while this works well for naturally occurring features such as hills and rivers, they cannot be directly applied to characters themselves. This project aims to create an application which can generate village populations with shared history and ancestry of approximately 100 years that could be either exported directly into a game world or used by a designer to act as a base for a population.

1.2 – Project Focus

The project focuses on finding what makes a believable NPC and then attempting to replicate that believability through procedural generation.

1.3 – Project Aim

The aim of this project is to create an application which can procedurally generate the population of a village by simulating a history. The NPC Villagers should feel connected to one another, yet each be distinctly different from each other.

1.4 – Objectives

The primary objectives of the project are:

- To research existing NPCs to discern what data makes up a good NPC.
- To create an application which can procedurally generate the population of a village by simulating a history for it.
- To critically evaluate the application and the data it outputs.
- To compare the data produced by the application against NPCs from existing and well received games.

1.5 – Research Question

Would Procedural Generation be a viable tool for creating large and engaging NPC populations?

1.6 – Ethics Considerations

This project follows Abertay's ethics guidelines to ensure that the risk of any ethical issues arising is kept to a minimum. The only ethics concern is the questionnaire; however, measures have been taken to ensure it adheres to the Data Protection Act 2018.

1.7 – Overview

Following this introduction, the dissertation will proceed in the following order:

- Literature Review – Outlines the current state of the art games using procedural generation for NPCs, as well as brief analyses of core scholarly sources referenced throughout the project.

- Methodology and Practical Application – Covers the three stages of producing the data to be later analysed; project design, practical application/implementation of code, and the methods used to record results from testing.

- Results – Visual representation of the results recorded from testing, with brief descriptions of data trends. Also includes a brief population growth analysis.

- Discussion – An analysis of the results recorded, as well as a critical review of the method for gathering and interpreting the data. This is followed by a similar review of the application itself.

- Conclusion – The final decision on the success of the project, as well as potential improvements for future work.

Chapter 2 – Literature Review

2.1 – State of the Art

Populating a virtual world is no small task. Even the smallest worlds like those found in Assassin's Creed games (Ubisoft Montreal, 2007), which can go as small as 1 square mile (Llewellyn, 2020) require dozens, if not hundreds, of NPCs to feel inhabited. The main goal of these NPCs is dependent on the genre of game. In an RPG the main purpose may be to guide the player through the story, or to simply be disposable fodder for the player to carve a path through, essentially viewing the NPCs as their avatar would. In a more detached simulation such as that found in Dwarf Fortress (Adams and Adams, 2006), the player can observe everything but has limited direct contact with the world. In these games the main role of an NPC will be to look convincing when observed by the omniscient player, with a fleshed-out history and/or personality to lend the world some depth. This is due to the player not observing the world from the perspective of a character within it, but rather as themselves playing a game.

For most RPGs, the example in this case being Oblivion (Bethesda Softworks, 2006), making a character who keeps the player engaged and immersed within the world requires the manual writing of a backstory and motivations (Mikkelson, 2017), just like in the pen and paper games that the genre originates from. The reason for this is that a believable agent should have identifiable social relationships between characters, and an individual personality (Gomes *et al.*, 2013), among other traits. These traits are something that most humans can recognise implicitly but are difficult to translate into an NPC, even if the character is created by hand (UESP, 2017), (Zero Punctuation, 2008). The complete inverse of this would be to generate all characters procedurally, which presents its own merits and drawbacks. Current examples of this include galactic scale games like Elite Dangerous (Frontier Developments, 2014) and No Man's Sky (Hello Games, 2016). Both are examples of a game's scale preventing a development team of any size having the time or tools to manually populate the 400 billion star systems as seen in the smaller of those two games

(EDW, 2021). Naturally this requires procedurally generated NPCs to make the space feel like the player isn't the only being in the galaxy. The natural trade-off when generating content procedurally as opposed to manually is that of quantity over quality when done wrong. This does not necessarily mean that the generation is poorly executed, simply that humans require a suspension of disbelief to fully regard the supernatural or romantic as real, regardless of characteristics out of the ordinary, with the term first being coined by English poet Samuel Taylor Coleridge (Coleridge, 1985). The issue with procedural generation lies in how easily this suspension collapses when presented with what is ostensibly a human NPC that communicates in a manner that is unmistakeably artificial.

Figure 2-1 Generated Conversation in Elite Dangerous (Frontier Developments, 2014)

This is a basic overview of NPC generation and its limitations at the current time. Generating a city's worth of fully realised NPCs is still a way off. However, individual aspects of NPC generation have had their boundaries pushed in recent years. Watch Dogs Legion (Ubisoft Toronto, 2020) is an open-world game set in near-future London where there are over 9 million NPCs in the world (Saksena, 2020). The way the NPCs are handled in Legion is that before the player interacts with an NPC they are simply a randomly generated citizen dressed appropriately for the area. Upon the player initiating dialogue with them the game will generate a name, personality and daily routine, among other traits, to create the illusion of them always having been a part of the world. From then on, the NPC will have a chance of appearing in an area that they have ties to and may be seen carrying out their duties. This is called the Census system (Jensen, 2020).

Taking an alternative route towards believable NPCs, Event[0] (Ocelot Society, 2016) has only one NPC which is a ship's AI. The player can only communicate with the AI by typing to it in the game. This AI can respond with over 2 million lines of dialogue (Matulef, 2016), but most importantly it sidesteps the issue of breaking immersion by being exactly what it claims to be; an AI which will play the game with you. This ability to convey character through just dialogue is what this project hopes to address.

2.2 – Core Sources

There are two main sources which will be discussed in this section for their studies of procedural NPC characteristics. The first of these is Vanya Eccles' Procedurally Generated Background Characters (Eccles, 2017), which focuses on an individual's day-to-day routine. The second is James Ryan's Curating Simulated Storyworlds (Ryan, 2018), which is a much more expansive overview of emergent narrative as a whole. Ryan defines emergent narrative as 'narrative that emerges out of computer simulation of character activity, or the methodology of generating narrative in that way' (Ryan, 2018 p.686). There are also several smaller papers which warrant further discussion, such as Metrics for Character Believability in Interactive Narrative (Gomes *et al.,* 2013), and How Much Time Does a Farmer Spend to Produce My food? An International Comparison of the Impact of Diets and Mechanization (Ibarrola-Rivas, Kastner and Nonhebel, 2016).

2.2.1 - Procedurally Generated Background Characters

This project comprises mainly of an attempt to make it possible for NPCs to act as individuals on a moment-to-moment basis with consistency. This consistency is rooted in the Five Factor Model (FFM), which Eccles asserts is one of the most successful attempts to 'develop a comprehensive taxonomy of human personality traits' (Eccles, 2016, p.28). The FFM measures personalities based on Extraversion, Agreeableness, Conscientiousness, Neuroticism and Openness to new experiences (McCrae and John, 1992). These factors are convenient for the simulation

of a virtual personality. They can be quantified using only 5 variables while still having a track record of correlations between traits and socioeconomic outcome, as documented in several journals (Roccas *et al.*, 2002), (Borghans *et al*., 2008). As well as these 'psychological' aspects of an NPC, each NPC was also able to keep track of its physical and emotional needs in a manner similar to The Sims (Maxis, 2000). These would combine with global variables such as the time of day or the weather to offset the decisions made based on the FFM. This leads to what would otherwise be a straightforward system, where the actors repeat the same pattern of activities that best fits their FFM outputs, becoming a complex and adaptive system as external forces can cause the actor to react in a reasonable manner to almost any circumstances.

Part of the effectiveness of Eccles' implementation is the Action Hierarchy, which ensures that an actor can only switch to actions similar to the one currently being performed. This improves both the believability of the actor's actions, as well as the efficiency of the code. The example given is that once an actor has evaluated that it is hungry it won't need to then refer to its hunger again when choosing what to eat once it has already undertaken the 'Have a Snack' action (Eccles, 2016, p.34). This hierarchy also helps to maintain a natural flow where actors won't go suddenly from combat to relaxing, only to return to combat after a few moments. Further application of these principles would allow for an actor to "learn" a new job or skillset by replacing its existing actions with another set. Taking inspiration from The Sims (Maxis, 2000) this could be taken a step further using Smart-Object-style architecture, whereupon entering a location the actor would temporarily gain access to actions associated with that location. The example given is to allow an actor to 'Start a Brawl' or 'Sing Boorishly' while they are in a tavern, but have those actions revoked once they exit the area (Eccles, 2016, p.34).

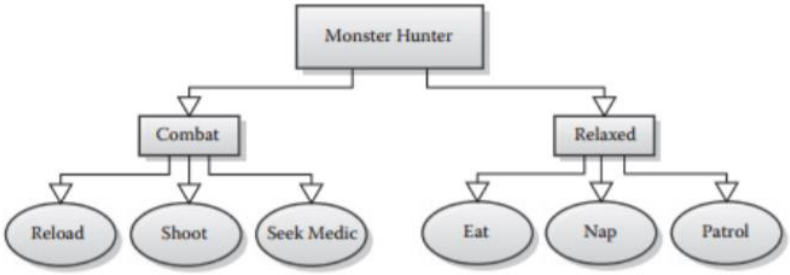


Figure 2-2 Example Tree Structure for Action Selection (Eccles. 2017 p.34)

The final notable feature of Eccles' project shown is the Interruption System (Eccles, 2016, p.55). This allows for certain entities in the world to trigger an interrupt event upon coming into contact with an actor, allowing them to consider whether they will continue with their current task or divert their attention to a new action. For example, the Socialise action has its own micro-hierarchy, so if one actor were to bump into another, they would check their current sociability and their current relationship with the other. If the utility of the current action is deemed to be less than that of socialising with a particular character then they will signal this to the other actor, who also has to accept to commence the Socialise action. Interrupts can of course be used for multiple other scenarios which require an element of spontaneous action, such as combat encounters. All of these elements layer upon one another to create a complex final actor with a CPU usage that scales linearly, allowing for reasonable assumptions about processor usage to be made in advance of running the program.

2.2.2 - Curating Simulated Storyworlds

This paper was recommended by Tarn Adams, co-creator of Dwarf Fortress (Adams and Adams, 2006), specifically pages 371 – 460; Chapter 9. This chapter focuses on the creation of a prototype game, Talk of the Town, where the player must use a network of resident gossips to find the identity of a secret NPC. The important element of Talk of the Town for this project is the generation of its population and world for the game, which is done by simulating the growth of a randomly generated town set in 1839. Every aspect from the years certain businesses will establish or collapse, to the names of children are all based on data from the USA (Ryan, 2018,

p.380). The simulation runs until August 19th 1979, with only a few major timesteps per year being simulated up until this point. Once the simulation reaches the termination date, the game generates the character knowledge required for the gameplay to function.

For the purposes of this project, however, the generation of the content is the key feature and is detailed thoroughly in Ryan's paper. Initially, citizens of the town are created randomly and fulfil the required roles such as farmers, miners and gravediggers. This random generation includes a full-face generation system that allows for children to inherit from their parents to ensure continuity. The names given to the NPCs are based on either their ancestors' names, which have a set probability of being used, or census data from a year appropriate to the one they are born into. While the world and population are being generated the simulation only does 4 days of each year to save computational power. To compensate, any interactions that occur during these days have their outcome values multiplied by the number of skipped timesteps since the last simulated timestep to account for this. Overall, Ryan's paper covers the logistics of containing and managing all of the appropriate data for simulating virtual lives, which makes it ideal reference material for this project.

2.2.3 - Metrics for Character Believability in Interactive Narrative

This short paper (Gomes *et al.*, 2013) is used to define believability metrics using perceived believability dimensions. The proposed dimensions are; behaviour coherence, change with experience, awareness, behaviour understandability, personality, visual impact, predictability, and social and emotional expressiveness (Gomes *et al.,* 2013, p.3). For the purposes of this project the focal points will be personality, social, and to a lesser extent, awareness.

The paper also proposes that the best way to analyse these metrics is to present a statement about a character and ask a participant to rate the statement on a scale of "totally agreeing" to "totally disagreeing".

The applicable templates are as follows (where X is replaced with character name):

Personality – < X > has a personality
Social – < X > interacts socially with other characters
Awareness – < X > perceives the world around him/her

2.2.4 - An International Comparison of the Impact of Diets and Mechanization

This study calculates the hours of farm labour required to produce one person's annual food consumption. The study covers four scenarios, with the one later referenced in this paper being 'Scenario 3: Basic diet with non-mechanised system' (Ibarrola-Rivas, Kastner and Nonhebel, 2016, p.8). The study is thorough in that it covers both large-scale farms and smallholders, as well as taking into account the amount of feed required to produce each kilogram of final product. For mechanised crop productions the data is gathered from the USA and Iran, and for non-mechanised the data is from India and Mexico. The assumed working day of the study is 8 hours, with Scenario 3 taking a total of approximately 350 hours to produce the food for one person. For the purposes of this paper, the total time taken to produce the food for one person per year has been increased to 400 hours. This is to account for possible crop diseases slowing pre-industrial food production.

2.3 – Concluding Remarks

The papers considered above were all selected due to their relevance to various aspects of the project. The first two, Procedurally Generated Background Characters (Eccles, 2017) and Simulated Storyworlds (Ryan, 2018) were chosen for their insight into the development of NPC generation systems, and were more integral to the design stage of the project. Metrics for Character Believability (Gomes *et al.,* 2013) was selected due to it succinctly covering ways to quantify NPC believability. This is essential in creating a balanced comparison of NPCs from disparate sources. Finally,

the study on the Impact of Diets and Mechanization (Ibarrola-Rivas, Kastner and Nonhebel, 2016) was required to ensure that the simulated world created had a grounding in reality. This grounding lends a level of believability that would otherwise not be present and raises the overall quality of the simulation.

Chapter 3 – Methodology and Practical Application

3.1 – Design

The design of the application was split into two main parts. The first of these is the market research stage. This involved scrutinising games that generated their characters to have personality, such as Ruinarch (Maccima Games, 2020), Rimworld (Ludeon Studios, 2018) and Dwarf Fortress (Adams and Adams, 2006). From observing these games a list of variables to be potentially tracked was constructed. The second step was to research how some of these games worked, as well as how some academic papers handled the subject like Curating Simulated Storyworlds (Ryan, 2018). This allowed the concepts developed in the market research stage to be refined.

3.1.1 – Market Research

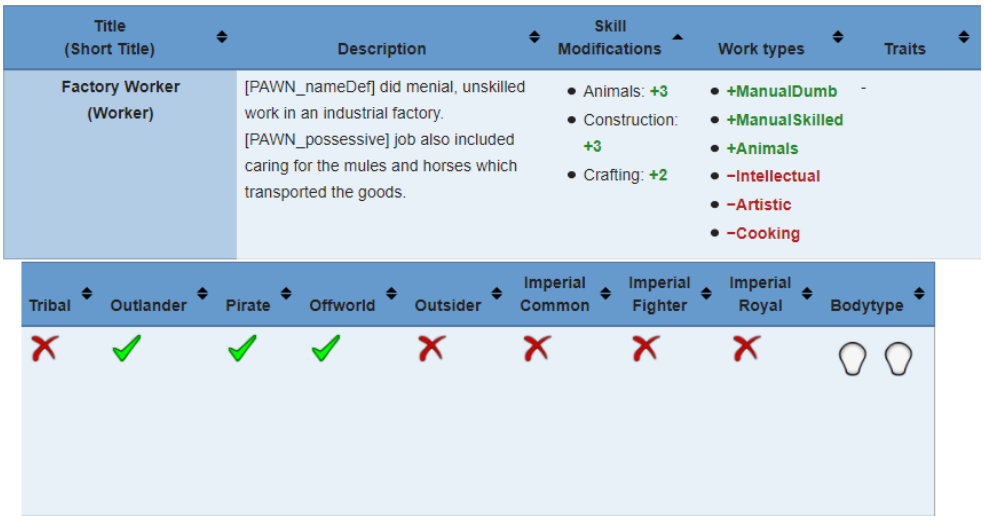
To ensure that the project was able to generate NPCs that would be comparable to those from existing games research was essential. The games which were primarily focussed on for this purpose were mostly under the ‘God-Game’ or ‘Management’ genre as these have easily visible variables for most of the characters in their worlds.

For Ruinarch (Maccima Games, 2020) this research was limited to what was accessible through the game’s Villager interface panels.

Figure 3-1 Ruinarch Villager Interface Panels (Maccima Games, 2020)

The tab which contained the most useful information was the ‘Info’ tab, as it details the traits of the Villager. These traits are what set each Villager apart from one another during gameplay, essentially acting as their Personality for the purposes of Metrics for Character Believability (Gomes *et al.*, 2013). The ‘Relations’ tab also adds depth to the Villager’s character by altering the way they interact with other Villagers. The main drawback of these values is that at the start of the game they are mostly neutral for those they share the village with, positive with family and negative with outsiders. This shows that the relations are not simulated before the game begins and does reduce the believability of the characters somewhat. Overall, Ruinarch has a well fleshed out Villager system which allows the Villagers to act in ways that are predictable and consistent with their character yet also react spontaneously to new situations. The main issue with using it for research was the lack of access to code, as well as a lacklustre Wiki, meaning that only surface level observations could be made from it.

Rimworld (Ludeon Studios, 2018) has an extensive Wiki dedicated to it, Rimworld Wiki (RW, 2018), which greatly aided in researching the inner workings of the game. Rimworld calls its NPCs ‘Pawns’, although this includes Humans, Animals, Mechanoids and Insectoids. The research for this project was only for Colonists, which is the controllable subset of Humans that the player interacts with. Most of the NPC generation in Rimworld is done through applying backstories to the Colonists. All generated Colonists will have a childhood backstory, of which there are 36, and those that are old enough will have an adult backstory, of which there are around 85. Colonists who age during gameplay do not have an adult backstory which can be used to distinguish a ‘naturally aged’ Colonist from a generated one.



Title (Short Title)	Description	Skill Modifications	Work types	Traits
Factory Worker (Worker)	[PAWN_nameDef] did menial, unskilled work in an industrial factory. [PAWN_possessive] job also included caring for the mules and horses which transported the goods.	• Animals: +3 • Construction: +3 • Crafting: +2	• +ManualDumb • +ManualSkilled • +Animals • −Intellectual • −Artistic • −Cooking	-

Tribal	Outlander	Pirate	Offworld	Outsider	Imperial Common	Imperial Fighter	Imperial Royal	Bodytype
✗	✓	✓	✓	✗	✗	✗	✗	

Figure 3–2 Rimworld Background Example (RW, 2018)

A Colonist's backstory determines their skills during gameplay, how they are perceived by other Colonists, and can enhance or impair their ability to complete certain tasks. Their backstory also determines their body type, which dictates how they are visually represented during gameplay. Upon generation, a Colonist's skills will range from 0 – 5 before their background is applied, however this is more for the gameplay than for the depth of the Colonists personalities. The backgrounds system is excellent for making many Colonists with skills that match a stated history, however it has a few flaws that stop it from being usable for creating large numbers of varied NPCs. The largest issue from a gameplay perspective is that child and adult backstories can sometimes clash. This may result in skills that a child background promotes being undermined by an adult backstory. This issue also means that a Colonist's backstory may clash thematically which is a problem when trying to generate a consistent and believable backstory. Backgrounds will also only refer to the Colonist whose background it is, meaning Colonists don't feel connected to other pawns through their backstories at all. Overall, the background system works for Rimworld, which is partly due to the premise of the game being about a rag-tag band of survivors as opposed to an established group. This would not translate to a successful generation of a larger group or village however. This is because backgrounds would either be too similar to one another to add

significant depth, or would be too different to be believed to originate close to one another.

Dwarf Fortress (Adams and Adams, 2006) was the single largest source of inspiration for this project. Due to the game being designed to be easy to modify, a lot of information on the game is easily accessible via the extensive Wiki for the game (DFW, 2007). Almost every aspect of Dwarf Fortress' worlds and characters are generated before gameplay begins. This ranges from which gods will make up a pantheon down to what the name of an ancient scroll is. The main research focus for NPC generation was the titular Dwarf race. As with Rimworld however, Dwarfs fall under an umbrella term for creation purposes, in this case being considered a creature. In Dwarf Fortress a creature is 'any animate, normally-mobile…being that can interact with the world' (DFW Creature Page, 2007). A creature can be further subdivided into castes, which for the Dwarfs is limited to Male or Female, however the basic template for all creatures within a caste is the same. The data which adds character depth to a Dwarf is called its Attributes, which govern how adept it will be at most tasks and is split into Body attributes and Soul attributes. Body attributes are used to quantify anything physical about a Dwarf, such as how agile or disease resistant it is, whereas Soul attributes measure more abstract concepts like intelligence and creativity. The depth from attributes is excellent from a gameplay perspective, but limited in how it displays distinct personalities.

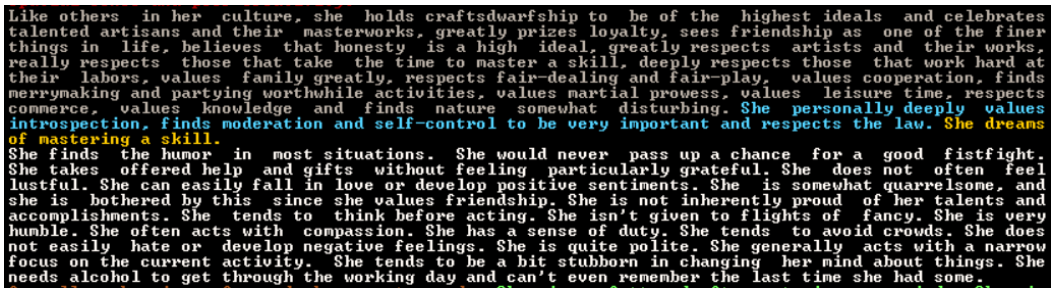


Figure 3–3 Dwarf Personality (Adams & Adams, 2006)

The real individuality of a Dwarf is found by looking at its Mannerisms, Beliefs, Goals, Facets and Memories. The most surface level of these is Mannerisms, which adds a superficial description to a Dwarf. Beliefs, Goals and Facets are considered to be a Dwarf's Personality Traits, and can greatly influence how a Dwarf behaves. For instance, a Belief could relate to a Dwarf's views on humour or peace, with the value of that belief informing their interactions with other Dwarfs. Beliefs can be influenced by the civilisation a creature grew up in, with Dwarfs holding craftsmanship in high regard. Facets are similar in that they inform a Dwarf's growth; however, Facets promote or stunt the development of certain skills and can clash with Beliefs. Facets can also be more prominent in a species, with a Dwarf's median 'Greed' being 55 on a scale of 0-100. Goals are more limited in types than Beliefs or Facets, only setting a goal which a Dwarf wishes to fulfil. These goals range from having a child to wanting to become immortal, giving each Dwarf a more tangible goal than to simply survive as might otherwise be the case. There are 33 Beliefs, 13 Goals and 50 Facets a Dwarf can have thoughts on (DFW Personality Traits, 2007). This would mean that, assuming each Belief or Facet was only binary as opposed to the 100-point scale that they are, there would be a minimum of 21,450 unique Dwarf combinations, assuming only one Belief, one Goal and one Facet. The data that elevates Dwarf Fortress' generation above other games when it comes to the believability of NPCs is Memory.

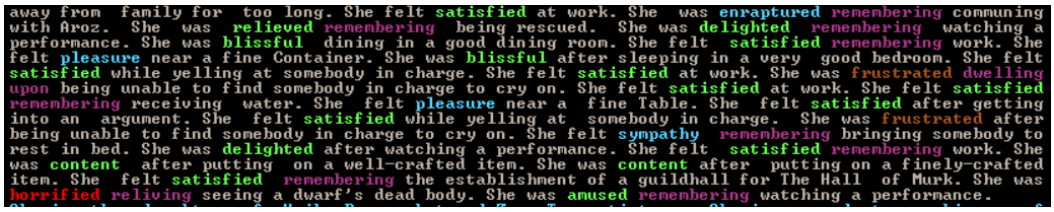


Figure 3–4 Dwarf Memories (Adams & Adams, 2006)

A Dwarf has 3 types of Memory; Short-Term, Long-Term and Core, with Short and Long-Term Memory each having 8 slots for possible memories (DFW Memories, 2007). Any 'thought' a Dwarf has can be added to Short-Term Memory if no existing Memory of that category exists, or if it has more 'emotional value' than the weakest Memory currently stored. After a

Memory has been Short-Term for a year it will attempt to be added to Long-Term Memory by a similar process, leaving a space in Short-Term Memory to be filled regardless of whether the Memory successfully becomes Long-Term. A Core-Memory has a 1:3 chance of being formed whenever a Long-Term Memory is thought about, and upon creation can permanently change a Dwarf's personality based on the systems discussed earlier. As with Short to Long-Term Memory changes, if a Memory becomes Core it is removed from a Dwarf's Long-Term Memory. Memories are a powerful tool for adding believability to the NPCs in Dwarf Fortress, as it allows influential events from an NPC's past to inform their current mood or course of action. This is an excellent method for adding growth to an NPC, as well as fleshing out their history in a way that is accessible to a player. Overall, the Dwarfs are exemplary NPCs, however, they are far too complex to recreate for the purposes of this project and as such only some key features will be adapted.

3.1.2 – Design Refinement

Deciding on a set of key variables to be tracked and developed over the course of a simulation based on the above research was the next step. There were only a few variables that all of the aforementioned games use, those being Name, Gender, Job, Friends/Relationships with others and whether they are Alive or not. Ruinarch does not track the Age of its villagers or how they are injured, although both are excellent sources of extra character depth. Looking at Curating Simulated Storyworlds' section on Character Personality (Ryan, 2018, p.384) it was also apparent that some equivalent personalities to Dwarf Fortress would be useful. The chosen personality model for this project is the Myers-Briggs Type Indicator (M&B Foundation, 2021). This was because of its widespread use and easy comprehension, as well as its ability to be stored effectively as four Boolean values. By the Proposal stage of the project the following table of variables had been assembled, however it is important to note that this table was not final.

Table 1 – Proposed Variables

Baseline	Name	Variable Type
Yes	Forename	String
Yes	Surname	String
Yes	Gender	Bool
Yes	Job	String
Yes	Alive	Bool
Yes	Location	String
Yes	Myers-Brigg Type	Array of Bools
Yes	Parent Male	Array Pointer*
Yes	Parent Female	Array Pointer*
Yes	Partner	Array Pointer
Yes	Age	Integer
Yes	Kids	Array of Pointers
No	Injuries	Bool/String
No	Friends	Array of Pointers
No	Local Wars^	Bool/String
No	Distant Wars^	Bool/String
No	Current Diseases^	Bool/String
No	Natural Disasters^	Bool/String

*Second generation onwards

^At a "World" level, may be referenced by NPCs

Table 3-1 Proposed Variables

3.2 – Implementation

To create an application to simulate a village population of individual NPCs the chosen language was C++. C++ was chosen as the Object-Oriented nature of the language would work well for keeping data encapsulated, with a Villager class making data handling much more intuitive. Making the application exclusively using C++ with only basic libraries would also reduce the size of the application, reducing compile times when compared to using an engine, as well as reducing the time it would take to create backups. The completed code for the project is in two main sections, the Main Body of the code and the Villager class code. There is also some unused code for simulating neighbouring villages, however this could not be completed to a polished standard due to time constraints.

3.2.1 – Main Body of Code

The Main Body code handles all of the initialisation of the simulation, including loading names and receiving user input. It also handles outputting results to a file.

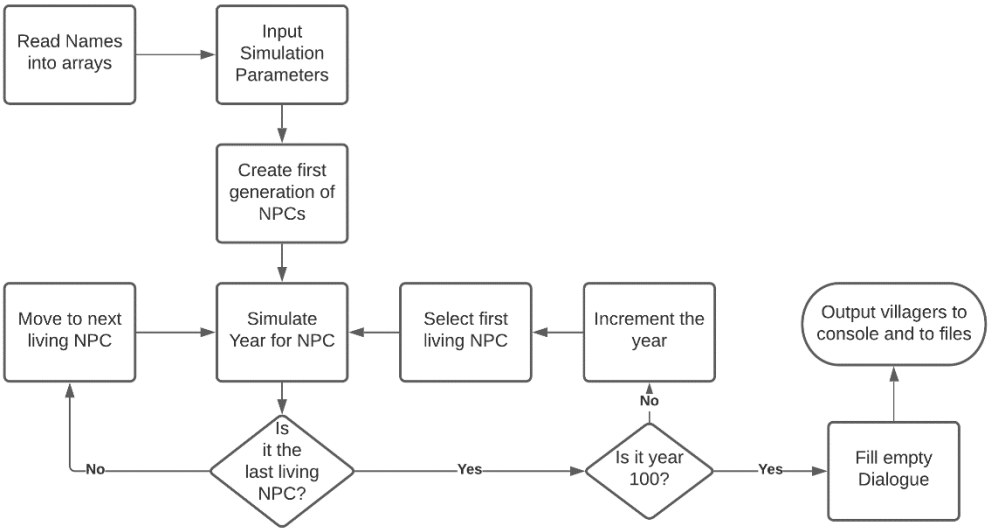


Figure 3-5 Main Code Flow

3.2.1.1 – Initialise

The first step upon running the program is to prepare to run the simulation. To give the NPCs a diverse range of forenames and surnames .csv files of male forenames, female forenames and surnames are loaded into corresponding arrays. The data for these files were sourced from modified versions of files taken from the National Records of Scotland (NRS, 2021). Currently these arrays are static, and as such only the current files will work correctly. There are 240 male names, 327 female names and 2088 surnames. Once the names are loaded the user is prompted to input values to be used during simulation.

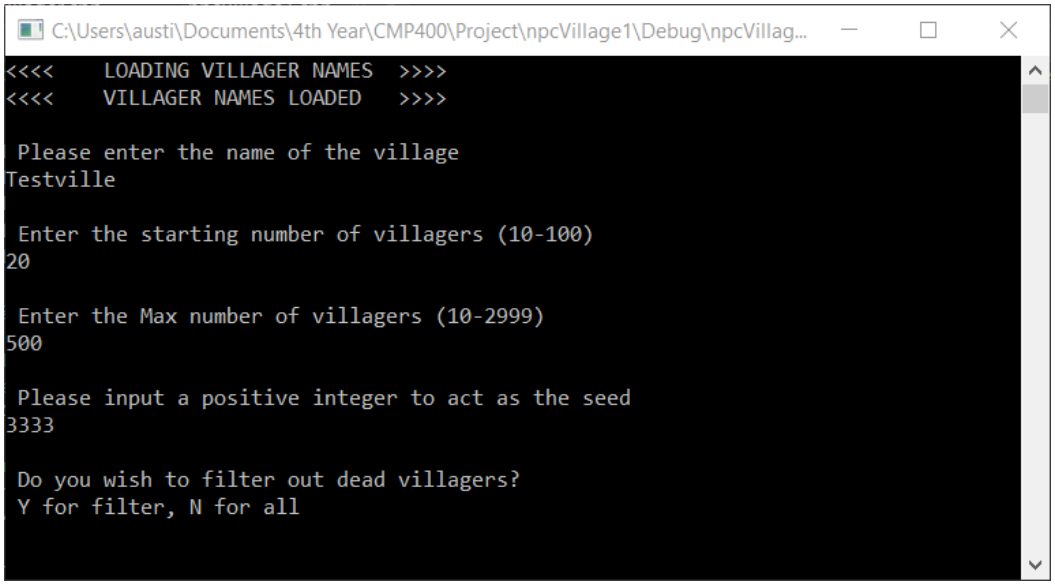


Figure 3-6 Inputs Example

The values the user has control over are: Village name, starting number of Villagers (between 10 and 100), maximum number of Villagers (between 10 and 2999), the seed for the generation/simulation, and whether the output should remove dead Villagers. The name of the village will be automatically capitalised when used during simulation and output. The starting number of villagers is fairly important as there are certain seeds that will die out if the starting population is too small. Currently the maximum number of villagers doesn't have any effect on the simulation, however it is designed to allow for the simulation to run until the population reaches the maximum number as opposed to the default 100-year simulation. The seed allows the user to replicate previous simulations as the seed determines the values of any numbers generated during the run of the program. This means that if a user liked the names of NPCs generated by a 15 Villager simulation and wanted more of the same, they could run the simulation with the same seed but a higher number of starting Villagers.

After user input is received the first generation of Villagers is generated. The first villager generated, Villager[0], is always named Fimbultyr (Sturluson, 1220). Fimbultyr exists to act as both parents to the first generation of Villagers to avoid having NULL values where that data would

otherwise be. Once Fimbultyr exists the Villagers are generated, with half of the first generation being male, and half being female. If the number is odd then an extra female Villager will be generated. New Villager generation will be discussed further in section 3.2.2. Villagers from generation one have set values for a few of their variables, notably those relating to age. Generation one Villagers will all be 20 years old, born in the year -20, as well as being marked as the first generation. Generation one chooses its surnames at random from the 2088 available, and is currently the only point at which new surnames can enter circulation.

3.2.1.2 – Simulate

Once the first generation has been created the simulation begins. A simulation currently lasts for 100 years, with each step taking up 1 year. As seen in Figure 3-5 the simulation works as a nested loop where once all living Villagers have been simulated for the year, the year will progress and output text to the console to confirm this. This output will display which year is currently being entered, and how many Villagers are currently alive. The details of the simulated year are handled by the Villager class, and as such will be elaborated upon in section 3.2.2. Upon the completion of a simulation all Villagers that don't have all 5 of their possible dialogue options filled have dialogue added to ensure that the Output stage has sufficient material to present.

3.2.1.3 – Output

There are two main forms of output from the program; to the console and to a file. The console output is mostly for debugging purposes as it displays notable activities that occur during a simulated year such as injuries, weddings and deaths. The console also outputs a large portion of Villager data in an easy-to-read manner for similar debug purposes, and the seed used to generate the villagers is output at the end.

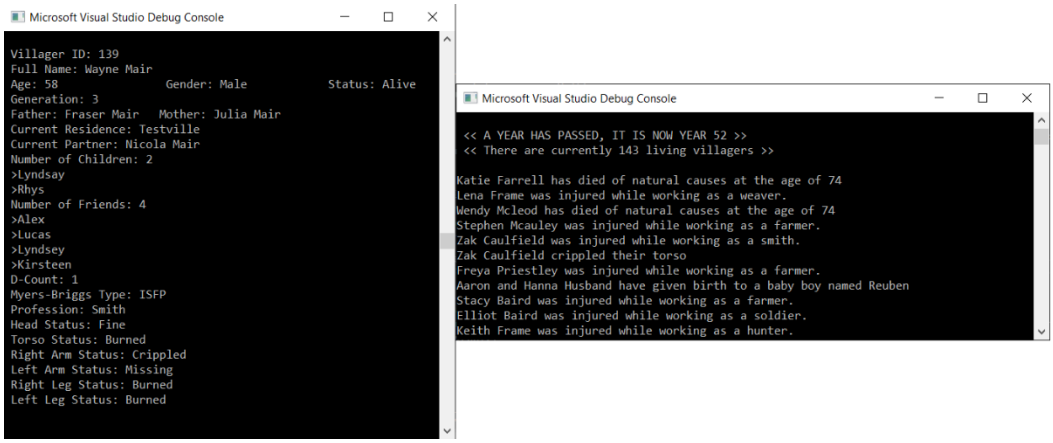


Figure 3-7 Console Outputs

The primary output is a .csv file of the results of the simulation. The name of the file is derived from the starting number of Villagers, the village name, the village seed, and whether the dead Villagers are included or not. For example, a village called “Murton” which started with a population of 15 Villagers, had a seed of 1234 and had the results filtered would create a file called “15Murton1234filtered.csv”. This means that it is possible to recreate any simulation using the information contained in the file name. The information contained in the .csv is in Table 3-2.

Variable	Data Type	Variable	Data Type
ID Number	*Integer*	No. of Kids	*Integer*
Forename	*String*	No. of Friends	*Integer*
Surname	*String*	Partner ID	*Integer*
Birth Year	*Integer*	MBTI Type	*Boolean (x4)*
Sex	*Boolean*	Job	*Enum*
Age	*Integer*	Generation	*Integer*
Mother ID	*Integer*	Questions (x5)	*String*
Father ID	*Integer*	Answers (x5)	*String*

Table 3-2 Output File Values

The information output to the files is limited in its usefulness, and the data prioritised was dialogue and the context surrounding it for the purposes of testing. Currently it is possible to trace backwards through time using the

Mother/Father ID values to find a Villager's parents, however the reverse is not possible with the limited data in the file. The program keeps track of a Villagers children at runtime, however given time constraints it was not an effective use of time to implement a way to view these relationships in such a manner.

3.2.2 – Villager Class

The Villager class code handles the data for each villager, as well as simulating individual events for a villager.

3.2.2.1 – Villager Data

Each individual Villager is its own instance of the Villager class and as such the class must contain all of the data that is unique to a Villager.

Variable	Data Type	Variable	Data Type
idNumber	*Integer*	ParentF	*Villager*[1]
Male	*Boolean*	ParentM	*Villager*[1]
BirthYear	*Integer*	Friends[4]	*Villager*[1]
Generation	*Integer*	Partner	*Villager*[1]
Age	*Integer*	mbEI	*Boolean*
Alive	*Boolean*	mbSN	*Boolean*
isPregnant	*Boolean*	mbTF	*Boolean*
Forename	*String*	mbJP	*Boolean*
Surname	*String*	Head	*Wound*[2]
FriendCount	*Integer*	Torso	*Wound*[2]
Kid[6]	*Villager*[1]	ArmR	*Wound*[2]
KidCount	*Integer*	ArmL	*Wound*[2]
activeDialogue	*Integer*	LegR	*Wound*[2]
dialogue[5][2]	*String*	LegL	*Wound*[2]
dCount	*Integer*	Job	*Role*[3]
Location	*String*	deathRisk	*Integer*

[1] Pointer to another Villager

[2] Enum that can be Fine, Scarred, Burned, Crippled or Missing.

[3] Enum that can be Dead, Farmer, Smith, Hunter, Housewife, Weaver, Soldier or Child.

Table 3-3 Villager Data

While much of the data is self-explanatory a few warrant further discussion. The 4 variables that begin with mb are Booleans that collectively make up a Villager's MBTI which serves as their personality and their genes combined. Male is a Boolean because keeping track of only two sexes is as complicated as the simulation needs to be in that regard. The activeDialogue keeps track of which of the Villager's dialogue slots was last edited. This allows the previous entry to have extra text appended to the front or back of the string. Location exists so that if the code for external villages was completed a Villager would be able to remember their hometown properly. Role is an Enum to manage which job a Villager has. Any Villager below 16 is a Child, and a Villager's Role is set to Dead upon death to ensure that they are not counted during the Role assignment stage of a Villager when they reach working age. Wounds track the condition of a Villager's individual limbs, as well as their head and torso. There are multiple stages of injury, however the only ones that cause a serious instant reaction are crippling the head, or losing the head or torso, which results in death. Aside from death by natural causes, this is currently the only way for a Villager to die.

3.2.2.2 – Simulate Year

The primary function of the Villager class is the Simulate Year function. From this function almost all of the lesser functions are called in some form or other, with the exception of those designed specifically for data output.

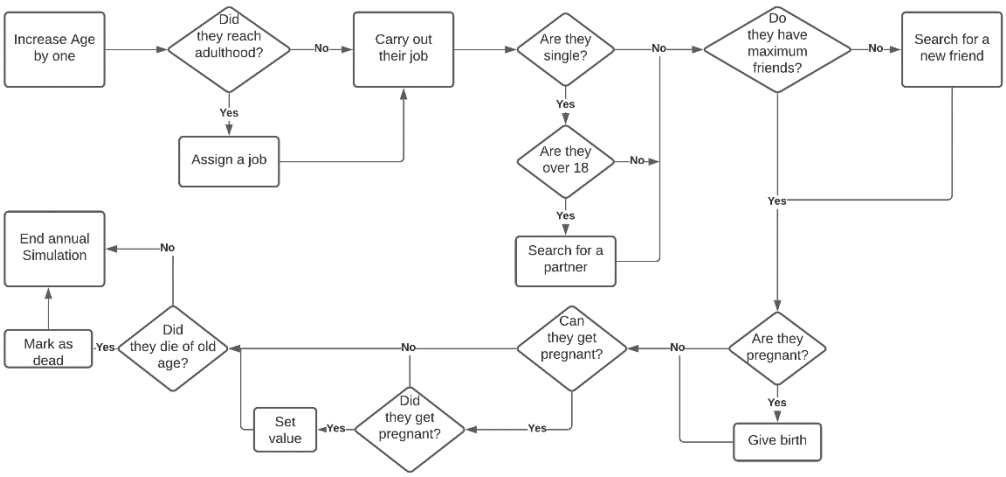


Figure 3-8 Year Simulation Flowchart

The Simulate Year function will only act upon Villagers who are alive to save memory and to avoid ageing dead Villagers. As shown in Figure 3-8 the function checks are prioritised in such a way that any functions which rely on having the most up-to-date information possible will run later. For example, a natural causes death is calculated based on risk that can be accumulated throughout the year, and is therefore checked last.

The requirements for running functions called by Simulate Year are also handled by this function. The most complicated of these requirements are those to become pregnant. To become pregnant the Villager must be Female and be married, although their partner cannot be dead. The Villager must also have a 'KidCount' smaller than the size of the 'Kid' array which stores who their children are. The final requirement is that the Villager may not be over 50 to account for Menopause (NHS, 2018). The least complex check is whether the Villager will have a child, which simply checks if the Villager became pregnant the previous year.

3.2.2.3 – Notable Functions

- Grow Up – This function triggers once a Villager turns 16 and assigns the Villager a job and some dialogue that relates to this assignment. Male Villagers can become Soldiers, Smiths or Hunters whereas Female Villagers can become Weavers or Housewives.

The function also ensures that 1 in 5 Villagers become a Farmer, as according to An International Comparison of the Impact of Diets and Mechanization (Ibarrola-Rivas, Kastner and Nonhebel, 2016, p.8) this is the minimum ratio required for a non-mechanised society to feed itself.

- Perform Job – All of the Villagers that are alive perform their job each year. A job is considered to be anything which takes up the majority of a Villager's time in a year, and as such 'Child' is considered a job. Each job has its own levels of risk involved, and this risk can be further affected by the MBTI type of the Villager performing it. Certain traits will make a job less risky, while others will make it more so. An example of this is that an Extroverted Farmer will be less likely to injure themselves at work, but an Extroverted Smith is more accident prone. The likelihood of injury is also increased by a Villager's deathRisk, meaning that the effects of a wound can often be cumulative. The levels of injury also vary, with Children unable to lose limbs at any point but Soldiers having a 30% base chance of minor injuries.

- Injure Villager – There are four specific types of this function; Scar, Burn, Cripple and Sever. The severity of a wound is determined during Perform Job. A wound can be applied to the head, torso or limbs, and if a Villager suffers a greater wound on a limb than already exists it will be replaced. When a wound is applied the Villager's deathRisk will be increased appropriately to reflect this, and risk will always be increased even if a greater wound already exists. A Villager may also be outright killed by a wound, with a crippled head, severed head or severed torso resulting in immediate death.

- Natural Causes Death – Aside from death from injuries, Villagers can only die from natural causes. This is the last part of a Villager's

yearly simulation so that it can be done with the most up-to-date information available. Death by natural causes is calculated by generating a random number between 70 and 170, which is then compared to the Villager's Age combined with half of their deathRisk. If this combined number is greater than the random number then the Villager dies. This means that a perfectly healthy 80-year-old Villager has a 10% chance of death by natural causes, however the vast majority of Villagers will have significant deathRisk by this point, ensuring deaths at appropriate ages.

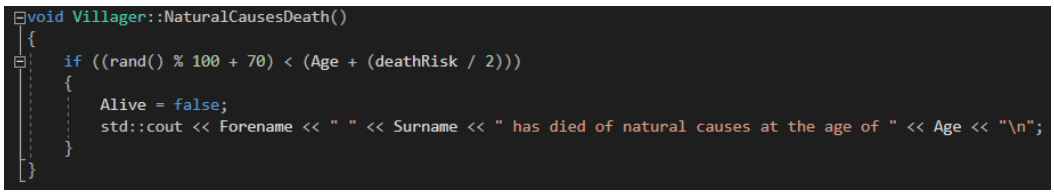


Figure 3-9 Natural Causes Code

- Have Child – An integral part of the simulation is reproduction. If the requirements outlined in section 3.2.2.2 have been met and a female Villager became pregnant the previous year, then they will give birth in the current year. First the function checks to see if the birth will have any complications. There is a 10% chance that a birth will result in the mother's death, and if this happens there is a further 20% chance the baby also dies. This means there is a 2% chance that both the mother and child will die. If the birth is successful there are several values a child will inherit from their parents. There is a 6.67% chance a child will be given the name of their paternal grandfather or maternal grandmother. Alternatively, they will be given a random gender appropriate name from the list. A child's MBTI is inherited from their parents where they share the types, and randomly assigned where they differ. If a father were to be an ENFP and a mother an ENTP, their child would be an EN[?]P. In this way the MBTI is treated like the genes of a Villager. Once a child has been generated there is a 20% chance that they will have dialogue added in which they mention the year they were born in.

3.3 – Recording Results

To be able to quantify the success of the simulation the output would have to be measured in a way that made it comparable to existing games. To achieve this the data that was best suited for comparison was the Villager dialogue. This dialogue would be compared with dialogue from a game with manually written dialogue and a game with procedurally generated dialogue. The manually written game was The Elder Scrolls IV: Oblivion (Bethesda Softworks, 2006), and the procedurally generated game was Dwarf Fortress (Adams & Adams, 2006). These were chosen for comparison as they were both released in 2006 yet are still relevant. In 2021, Oblivion still had a player count in the thousands (SteamCharts, 2021), and Dwarf Fortress is anticipating an updated release on the Steam platform (Steam, 2021).

To allow for a more accurate assessment the dialogue to be compared would all come from similar dynamics within their sources. The dynamic that was chosen was a family of three where all family members are still alive. From the project the simulated village was 100Newville1234filtered, the Cheung family comprising of Archie and Clare, as well as their daughter Maureen was chosen. The family from Oblivion was the Ottus family of Hastrel (ESW, 2018), Alessia (ESW, 2020), and their daughter Ida (ESW, 2019). The Dwarf Fortress family consisted of Tekkud Fathomrist as the father, Eral Atiskel as the mother, and Onget Tuloncerol as the child. As the Dwarf Fortress results cannot be easily replicated, a screenshot of the dwarf ‘brain’ of each family member can be found in Appendix A – Dwarf Brains.

The assessment was done in the form of a questionnaire which had several questions based on Metrics for Character Believability in Interactive Narrative (Gomes *et al.*, 2013). The questionnaire begins with a breakdown of how dialogue would be presented to the participant. The participant is then shown dialogue from one of the sources, but is not told what the source is. The participant is then asked 4 questions about the dialogue

where they answer by rating it on a scale of 1-5, with a higher score indicating that the dialogue is more believable. The questions are:

- How engaging is the dialogue?
- How involved in the world do these characters feel?
- How involved with the other named characters do these NPCs feel?
- How natural does the dialogue sound?

This is repeated for each of the 3 sources. At the end of the questionnaire participants are asked if they recognised any of the sources of data. This was to reduce contamination of the data by participants who may have preconceived notions about a game from playing it.

Images of the exact dialogue being evaluated can be found in Appendix B – Questionnaire.

A link to a copy of the questionnaire can be found in Appendix C – Link to Example Questionnaire.

Chapter 4 – Results

4.1 – Questionnaire Results

To ensure readability the data is presented in the same order as the questions in the questionnaire. A link to the complete table of results can be found in Appendix D – Results. There were 62 total participants.

4.1.1 – Oblivion Data

How engaging is the dialogue?

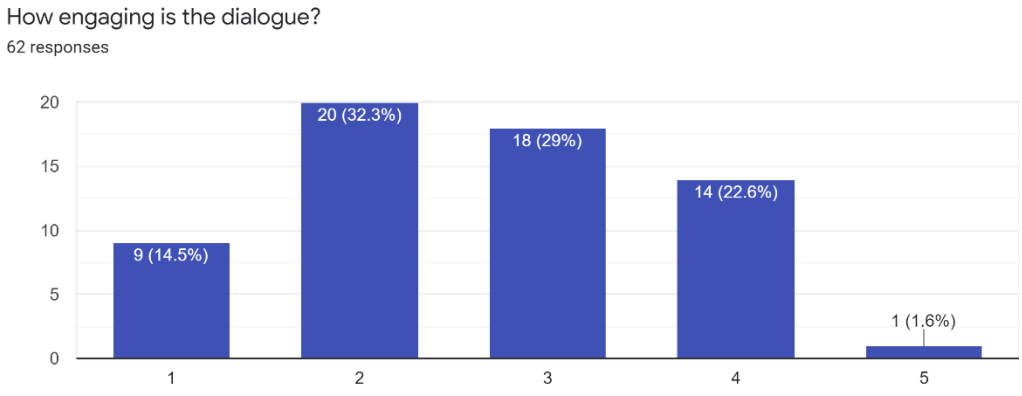


Figure 4-1 Oblivion Engagement

Average Rating: 2.64516129

Figure 4-1 shows that while the majority of participants rate the engagement to be at or below average, there is still a sizeable portion that found it suitably engaging.

How involved in the world do these characters feel?

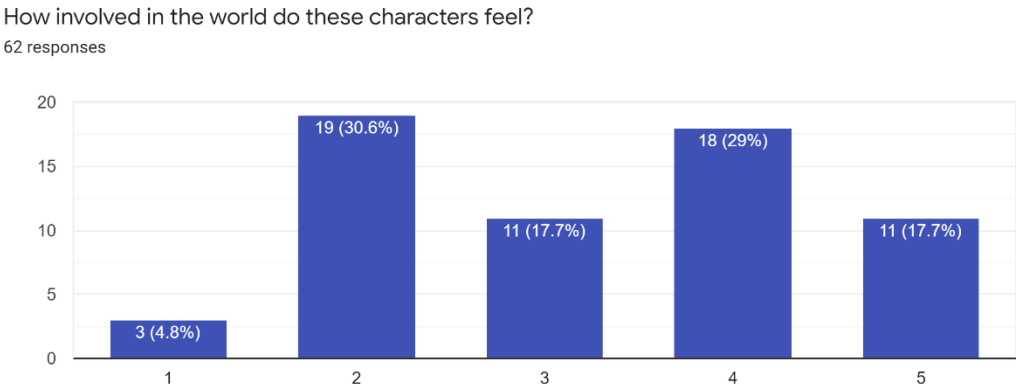


Figure 4-2 Oblivion Involvement in World

Average Rating: 3.241935484

While this question was fairly divisive it is notable that 40 of the 62 participants still rated the NPCs involvement in the world as being at or above average.

How involved with the other named characters do these NPCs feel?

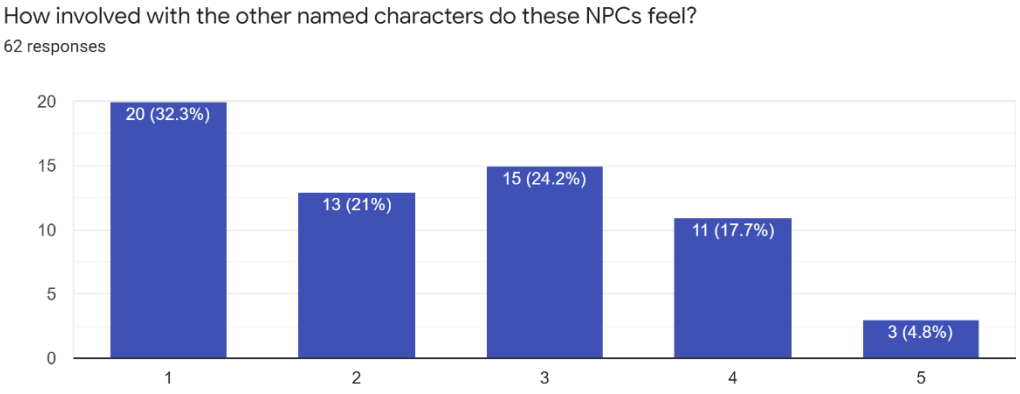


Figure 4-3 Oblivion Involvement in Characters

Average Rating: 2.419354839

Figure 4-3 shows clearly that the majority of participants viewed the involvement to be nearly non-existent, although there are still a few high ratings which raise the average.

How natural does the dialogue sound?

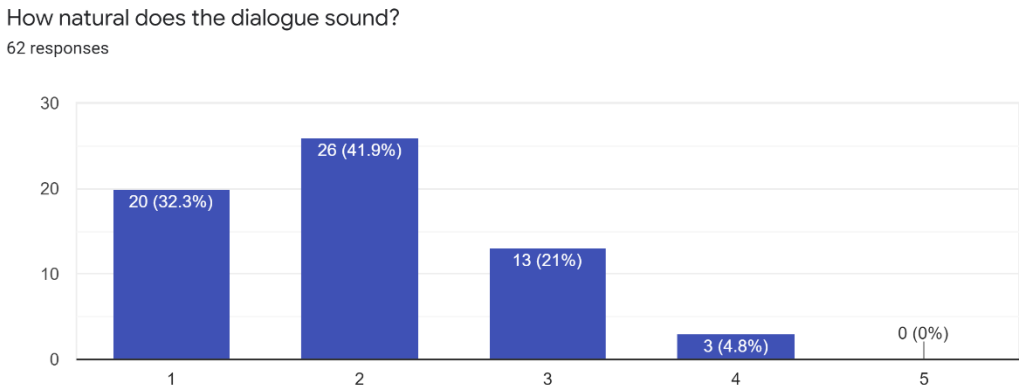


Figure 4-4 Oblivion Natural

Average Rating: 1.983870968

This is the only question with an average below 2, as well as the only question to not have a single rating of 5. There is a very clear trend towards the lower ratings, with less than a third of participants rating at average or higher.

4.1.2 – Village Data

How engaging is the dialogue?

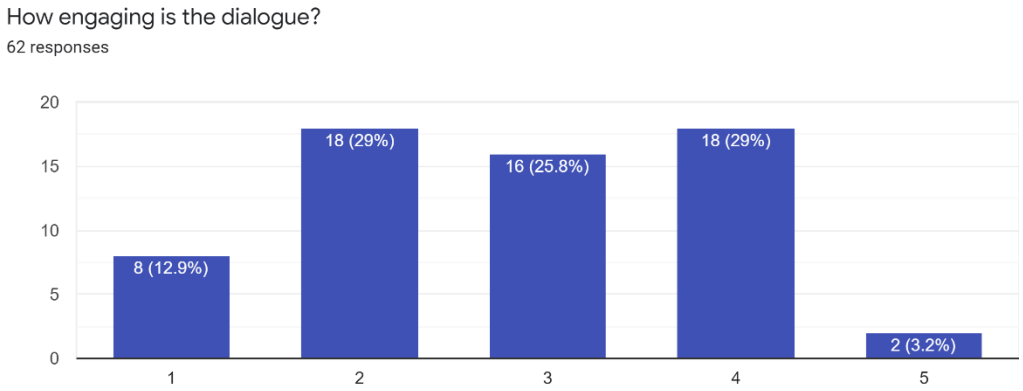


Figure 4-5 Village Engagement

Average Rating: 2.806451613

Figure 4-5 shows that the majority of participants viewed the dialogue as engaging, however there is a notable percentage that viewed it as not engaging at all.

How involved in the world do these characters feel?

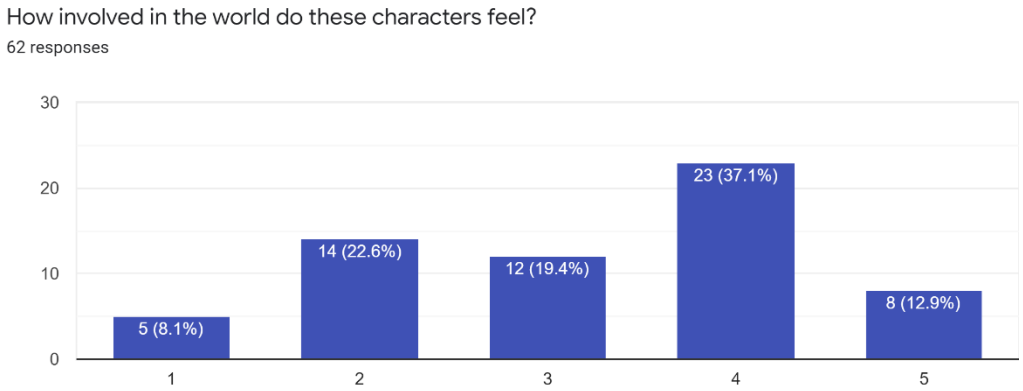


Figure 4-6 Village Involvement in World

Average Rating: 3.241935484

Although the results in Figure 4-6 are fairly evenly spread out there is a notable cluster of ratings at 4. It is also notable that there were

more ratings of the maximum value than that of the minimum, suggesting an overall trend to higher ratings.

How involved with the other named characters do these NPCs feel?

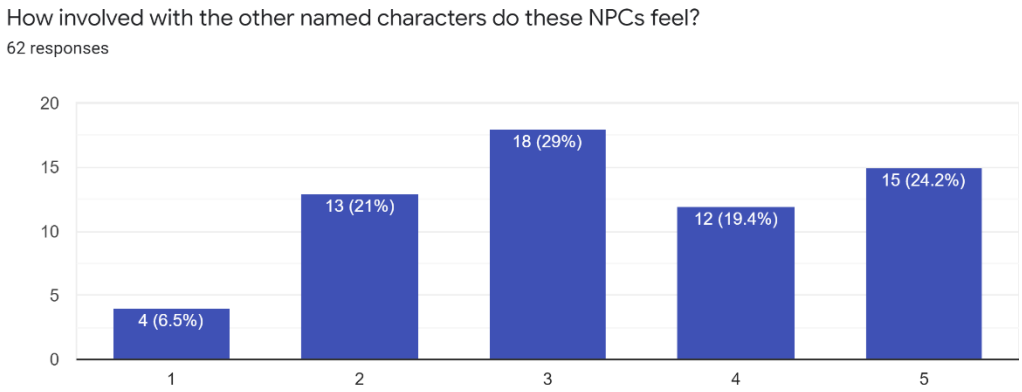


Figure 4-7 Village Involvement in Characters

Average Rating: 3.338709677

Less than a third of the ratings in Figure 4-7 are below average, which indicates that the NPCs felt like they had some form of relationship with one another.

How natural does the dialogue sound?

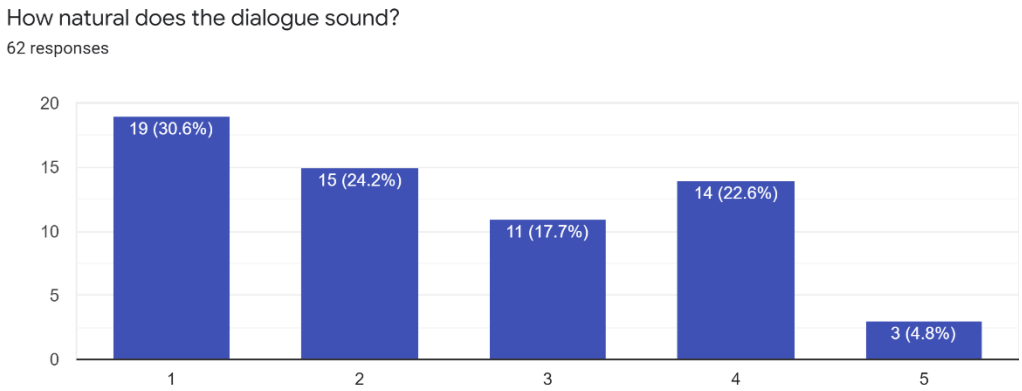


Figure 4-8 Village Natural

Average Rating: 2.467741935

In Figure 4-8 over half of the participants rated it as below average. Almost a third of the total ratings were at 1, which is the highest concentration of 1 ratings outside of Oblivion.

4.1.3 – Dwarf Fortress Data

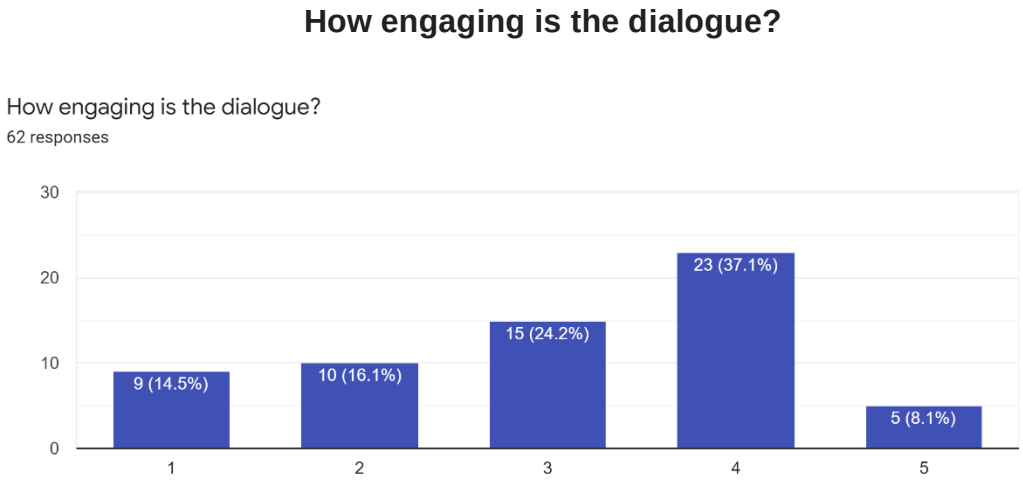


Figure 4-9 Dwarf Fortress Engagement

Average Rating: 3.080645161

Although the results in Figure 4-9 average out to almost a perfect average it is important to note that there is a higher number of ratings at or over average. This is then negated by a high number of 1 ratings which brings the average down.

How involved in the world do these characters feel?

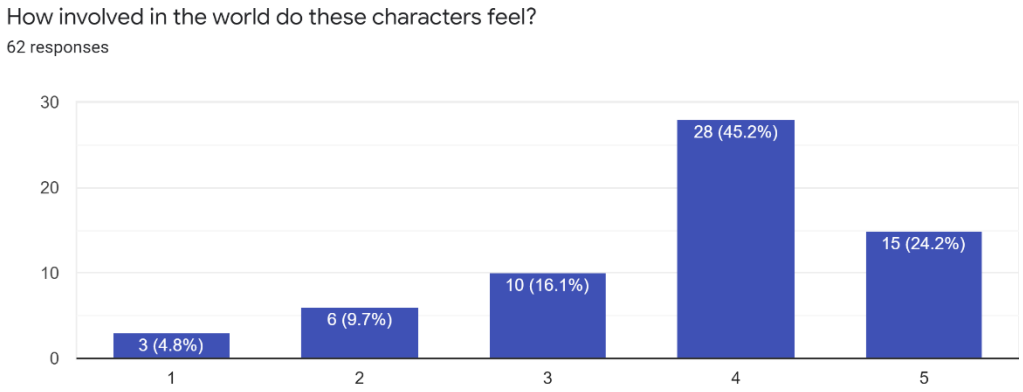


Figure 4-10 Dwarf Fortress Involvement in World

Average Rating: 3.741935484

Figure 4-10 has the highest average rating of all the questions. This is likely due to the fact that almost a quarter of participants gave this category the highest possible rating.

How involved with the other named characters do these NPCs feel?

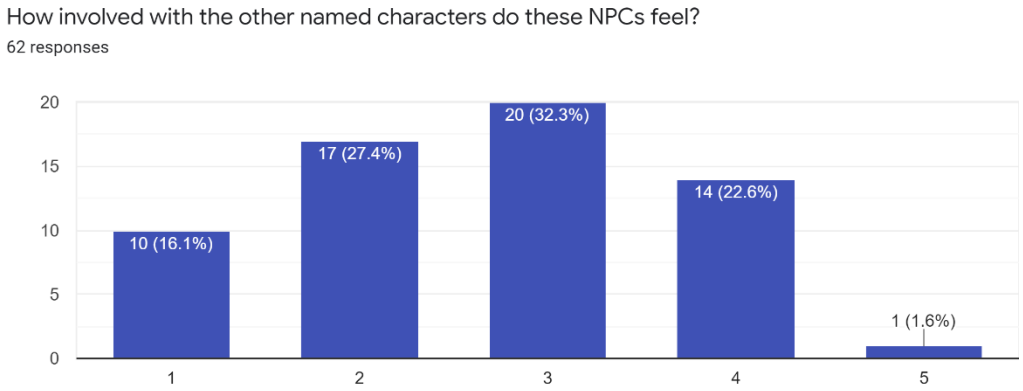


Figure 4-11 Dwarf Fortress Involvement in Characters

Average Rating: 2.661290323

Although Figure 4-11 clearly shows that the general rating of this category was fairly average at around 2-4 it is brought down significantly by a large number of 1 ratings.

How natural does the dialogue sound?

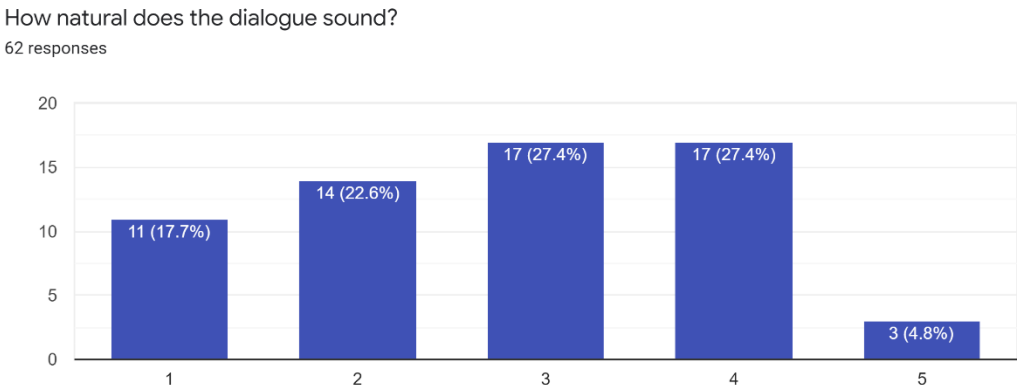


Figure 4-12 Dwarf Fortress Natural

Average Rating: 2.790322581

The ratings for Figure 4-12 are much like Figure 4-11 in that while the majority of the ratings fall within the 2-4 range, the average is brought down mostly by the difference between the number of ratings at 1 and 5.

	Oblivion	Village	Dwarf Fortress
Engaging	*2.64516129*	*2.806451613*	*3.080645161*
World	*3.241935484*	*3.241935484*	*3.741935484*
Characters	*2.419354839*	*3.338709677*	*2.661290323*
Natural	*1.983870968*	*2.467741935*	*2.790322581*

Table 4-1 Category Averages

Table 4-1 contains the averages for each category within the questionnaire to allow for faster comparison of larger trends. In this case a 3 is perfectly average, with anything above that implying a greater level of believability.

4.1.4 – Average Believability Scores

Shown below are the average scores of each source over all 4 questions. This can be considered to be a source's Believability Score.

Oblivion: 2.572580645
Village Project: 2.941110588
Dwarf Fortress: 3.068548387

4.1.5 – Data Contamination

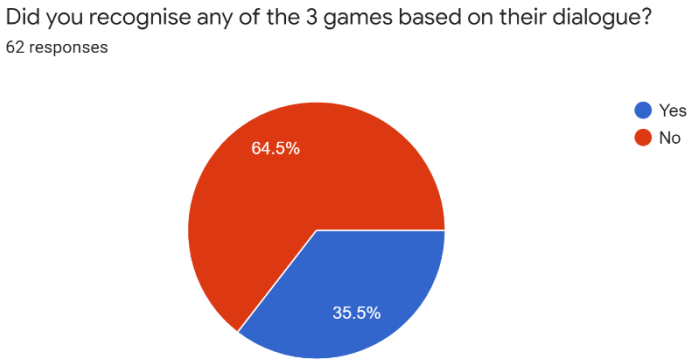


Figure 4-13 Data Contamination

Of the 62 participants in the questionnaire, 40 did not believe they knew the sources of any of the data. The average scores for those who did recognise the sources are as follows:

Oblivion: 2.352272727
Village Project: 2.81818181818
Dwarf Fortress: 3.136363636

All of these averages deviate from the overall Believability Score of a game by less than 0.25 points and so the contamination is minimal.

4.2 – Population Growth Analysis

Calculating the growth of the village population, the R number, allows for a prediction to be made for population size after time has passed. This means that should a user desire a population of a specific size they would be able to calculate the starting population and time required to reach that end population. The formula to calculate the R number is as follows:

$$PR = \frac{(V_{Present} - V_{Past})}{V_{Past}} x100$$

Where:

PR = Percent Rate
$V_{Present}$ = Present or Future Value
V_{Past} = Past or Present Value

Figure 4-14 R Number Calculation (University of Oregon, 2002)

The R number represents the annual population growth percentage. The number is difficult to calculate exactly due to the random nature of the simulation, however an average can be taken of multiple tests with varying population sizes. Table 4-1 has the R numbers for the seed '1234'.

Starting Population	R Number (%)
25	2.36
50	3.58
75	2.15
100	2.93

Table 4-2 R Number Calculation Results

An average R number can be calculated from these percentages, and comes out to be approximately 2.82%.

Chapter 5 - Discussion

5.1 – Overview

This chapter will be an analysis of topics discussed in previous chapters. Section 5.2 will focus on an analysis of the results shown in Chapter 4, as well as the effectiveness of the questionnaire method used to acquire the results. Section 5.3 will be an analysis of the project as a whole, including a brief overview of features that could not be implemented due to time constraints.

5.2 – Results Analysis

5.2.1 – Data Analysis

Due to the limited sample size of 62 participants, the results from the questionnaire will not be entirely conclusive. However, it is possible to look at the general trends that can be observed from the data.

The main goal of the testing was to try and determine if the dialogue output from the project was on a level of believability that would put it on par with existing games of a high quality. While it would be easy to say that the project's Believability Score was only lower than Dwarf Fortress' score by 0.127437799 and was therefore a success, this assumption by itself is flawed.

The Believability Score of the lowest and highest rated sources, Oblivion and Dwarf Fortress respectively, are only 0.496 points different. This would indicate that all of the sources were reasonably similar in their overall believability, however the difference between Oblivion and the Villagers was 0.368529943. This means that while the Villagers and Dwarf Fortress were fairly comparable based purely on Believability Score, Oblivion was rated much lower, in fact having the lowest score in every metric.

Oblivion's low score likely stems from the problem that procedural generation excels at fixing; having a large world that needs populated with

NPCs to make it feel alive. Based on the Believability Score it seems likely that as the NPCs from Oblivion were not involved with any quests, they were not given anything more than the most basic dialogue possible. This suggests that perhaps some form of dialogue generation for background NPCs could benefit open world games.

Focusing on the Villager project and Dwarf Fortress, there are much more noticeable differences in the ratings of individual questions. Table 4-1 clearly shows that while the project and Dwarf Fortress both have similar Believability Scores their individual question averages are distinctly different. The Villager project excels at making the NPCs feel involved with the world and the other characters within it, but lacks engagement and has dialogue that sounds fairly unnatural. Dwarf Fortress on the other hand excels at making the NPCs feel involved with the world, with it getting the highest score of any question. While Dwarf Fortress had the highest score of the 3 sources for natural sounding dialogue it still fell below average, likely indicative of the main issue with procedurally generated dialogue; the robotic tone of a lot of procedural outputs.

The one thing that the questionnaire data does clearly show is which areas of procedural NPC dialogue generation require the most work, both for the Villager project and for Dwarf Fortress. Natural sounding dialogue was the lowest scoring question across all of the sources, with none of them coming close to an average score of 3. Participant engagement was similarly low across the board, with only Dwarf Fortress reaching an average of 3. It is likely that if one of these scores were to increase the other would follow suit, however for the moment they are the Achilles Heel of procedural dialogue generation.

5.2.2 – Testing Method Analysis

The questionnaire method of testing the dialogue has yielded some interesting and insightful results, but the method is not without its flaws. The most noticeable flaw is caused by low sample size, where a single rating at a 1 or 5 can significantly lower or raise the average score of a

category. This is further compounded by another issue with the current form of the questionnaire. Each category can be rated from 1-5, with 3 being an average experience, however there is no way of conveying to a participant the exact nature of each category without potentially influencing the way that they vote. This means that an individual's perception of the value of a score can influence how they vote. Looking at individual submissions via the table in Appendix D there are definitely some participants who treated the values differently, with participant 9 being an example of a user who never scored any category above a 3.

Another issue with the questionnaire is the way in which the test data is conveyed to the participants. To make the information from the Dwarf Brains in Appendix A more easily readable it was reformatted to what can be seen in Appendix B. This may have influenced scores given by participants, but was necessary to reduce the number of participants who would be able to work out the source of the dialogue. This likely affected Oblivion in a similar manner, as although it has limited dialogue it is all voice acted within the game, and this may also have influenced player scores.

Issues also arise with how the data is best analysed once it has been collected. The method of using averages for each question allows for a quick comparison between categories in different sources, but averaging also erases a lot of the nuance that is visible in the graphs. Figure 4-9 demonstrates this well, with over 50% of the scores being at or over a 3, yet still only averaging out to around 3 due to the weight of 1 scores compared to 4 scores. This means the average score of 3.080645161 does not fully express that the majority of the participants rated at or above this score. Figure 4-2 and Figure 4-6 epitomise the loss of detail caused by the averaging system as both have the exact same average, yet the way the scores are dispersed are vastly different.

Some of the issues discussed have simple fixes in hindsight. The issues with questionnaire filling could be reduced by increasing the range of the

scores from 1-5 to perhaps 1-10. While this would make the form slower to fill out it would also allow for extreme scores, such as the current 1 and 5, to be used less frequently and encourage more moderate scores to be used. To further encourage this behaviour, it would also be prudent to have an example question which is prefilled to demonstrate the value of the points being used. This would likely increase the accuracy of the questionnaire as all participants would be working from the same baseline.

The issues involving the data averaging could also be alleviated slightly by using more than just the mean average for analysis, however this presents its own problems. Using the modal average would help to reduce the effects of instances where a large number of 4 ratings are negated by a small number of 1 ratings. The problems with this arise in instances such as Figure 4-2, where despite a majority of scores being 3 or higher the mode would be 2. Median average would potentially be a viable alternative, although due to the limited scale of 1-5 the results from this method would be limited. Overall, mean average is the most accurate option to hand, although any conclusions drawn purely from mean-averaged data should be thoroughly scrutinised.

5.3 – Project Analysis

5.3.1 – Methodology Review

When beginning the project the goal of creating a procedurally generated village population that felt plausible was clear from the start, however the method to measure the success of the project had not been decided. During the research stage a more concrete path to that goal began to form, with a clear idea of what each Villager would need to know about itself and others. Upon finding Metrics for Character Believability in Interactive Narrative (Gomes *et al.,* 2013) the decision to use an NPC's dialogue as a metric was formed. Had this paper been found earlier in the project's lifetime it is possible that the NPCs would have been less generalised in

their build, but instead geared towards generating more convincing dialogue. While it is interesting to speculate on such things it is likely that the project would have progressed in a similar manner, save for a higher prioritisation of some of the cut features which will be discussed in section 5.3.3.

5.3.2 – Application Review

The application for the project is fairly small and manageable, as the main priority while making it was to have it complete a single task well. Despite this there are still several improvements which could be made in the future. First of these would be to improve the way in which a new instance of the Villager class is created, as it is currently rather convoluted. Making this more readable in code would expedite future improvements to the application and would be a worthwhile investment of time. There is also a fair amount of legacy code in the project which has been left untouched due to the small size of the application, however now that it is in a working state it would be sensible to remove the data used to test the application in its early stages. Adding another way for the application to finish the simulation besides running for 100 years was always intended. The user inputs to allow for this are already accounted for with the Maximum Villager Count. The reason this is not already implemented is that there is a slim chance that a village may die out, and there is currently no way to account for this eventuality. The choice of output file could be improved upon in retrospect, with the .csv file that was chosen having one considerable drawback. The .csv file type means that Villager dialogue can never have commas in it, which drastically reduces the flow of dialogue. This means that the poor choice of file type had a negative influence on the overall believability of the dialogue, bringing the project down as a whole.

The application has had its fair share of bugs during development, although there is only one currently known bug. The bug in question is that first wave of Villagers do not generate truly randomly. This is not an issue in the grand scheme due to the random nature of events that effect the NPCs, as the seed will cause their lives to be wildly different regardless of how they start.

5.3.3 – Cut Features

Table 3-1 shows the proposed variables to be tracked by each Villager, but also shows variables to be tracked at a global level. This was dropped due to the focus shifting onto creating dialogue for the Villagers, as keeping dialogue more personal was intended to make it more believable to a user.

There is some unused code for ‘other towns’ which would have allowed for Villagers to arrive from them and move to the simulated village. This was cut due to the priority on dialogue and the fact that it would require the Villager creation system to be redone, thus taking up too much time. This in turn meant that the Maximum Villager Count method of ending the simulation had to be pushed back. This is the feature which would have most enhanced the application, as it would allow for new surnames and MBTI types to be introduced into a population that might otherwise stagnate, as well as making the world of the Villagers seem larger and more interconnected.

5.4 – Summary of Discussion

Overall, the project as a whole has run fairly smoothly. The main issues that have arisen have mostly been due to the application beginning development before a metric to measure the success of the project had been decided upon. This led to a fairly generalised application being developed first, with it only becoming specialised after the core Villager functionality was already present. While this was successful there would undoubtedly be a more believable dialogue output if that had been the sole intention from the start. While the results of the testing suggest that the project has been mostly successful due to the Believability Score being on par with established games, it is important to bear in mind the small sample size of 62 participants. More testing would be necessary to draw any conclusive results, both in terms of the number of participants as well as the variety of tests done. Surveying more games could also be a good way of increasing the accuracy of the tests, as with only two sources to compare to there are limits to the analysis that can be done. The project could

certainly have certain aspects streamlined, mostly in the code, but on the whole has been a success.

Chapter 6 – Conclusion

6.1 – Research Questions and Objectives

The results of testing, while not conclusive, do certainly suggest that the main aim of the project could be considered mostly successful. The main aim was: “to create an application which can procedurally generate the population of a village by simulating a history. The NPC Villagers should feel connected to one another, yet each be distinctly different from each other.” The relatively high Believability Score of the Village project combined with the above average rating of involvement with other named characters serves to confirm that the majority of this aim has been met. The only part of the aim which remains in doubt is the capacity for the generated NPCs to be “distinctly different from each other”, as this was never able to be tested for. Of the 4 objectives stated in section 1.4, all were successfully completed to a satisfactory level.

One unexpected outcome of the test results was the overall low Believability Score of Oblivion, which despite having manually written dialogue still scored the lowest in every category including how natural dialogue sounded. These results highlight the increasing need for some form of procedural generation of NPCs in open world games, even those of limited scale. For a game with a limited size, it would make the most sense to procedurally generate the population with an application similar to the one used for this project. This population would then have appropriate names, occupations, personalities and family units. Once this has been generated a designer/writer would only have to ensure that minor NPCs had generated properly, and add detail where necessary. This would save a lot of development time, especially if a generation system for physical appearances for NPCs was also added.

In regards to the research question of “Would Procedural Generation be a viable tool for creating large and engaging NPC populations?”, the answer is mostly yes. This project, which was designed as a prototype, has demonstrated the ability to generate large populations which have dialogue

which can make them feel like part of a larger community. The only aspect of the question which has proven difficult to fulfil is the engagement level of the generated NPCs. This is likely due to the small scope of the project which does not allow for the generated NPCs to interact with the user in a more meaningful way than answering predetermined questions.

6.2 – Future Work

In its current form the application has significant room for improvement, both in how it runs the simulation and in how the data is then output. Aside from adding the cut features from section 5.3.3 there is significant room for improvement in the way the program outputs data. Ideally, some form of XML based output which would allow for Villagers to be viewed as a web of family and friend ties could be implemented, although this would likely require a significant reworking of the application.

As discussed earlier in the conclusion, user engagement has proven difficult to provide via generated NPCs. This could be improved upon by increasing the scope of the project to include more user interaction. To achieve this the project would likely need to be put into some form of game engine like Unity to allow Villagers to have more forms of self-expression than just dialogue. The main downside to this is that the code may need to be converted from C++ to C# to accommodate, and the scale of the project may increase to an unreasonable level that would take too long to complete.

Once improvements have been made the next step would be to do all of the testing again, both with the original questionnaire format and with an improved format as described in section 5.2.2. This would allow the updated results to be compared against the current results to check for an improvement, as well as taking full advantage of the increased depth of the improved testing. While these future plans would take significant time to implement, the potential time saved from creating a fully functioning application could save countless man-hours in the long term, making this project well worth the time invested.

List of References

Adams, Z. and Adams, T. (2006) *Dwarf Fortress* [Video game]. Bay 12 Games.

Bethesda Softworks (2006) *The Elder Scrolls IV: Oblivion* [Video game]. Bethesda Softworks.

Borghans, L., Duckworth, A.L., Heckman, J. J. and Weel. B.T. (2008) 'The Economics and Psychology of Personality Traits', *Journal of Human Resources,* 43(4), pp. 972-1059.

Carpenter, L.C. (1980) *Computer rendering of fractal curves and surfaces. In Proceedings of the 7th annual conference on Computer graphics and interactive techniques.*

Coleridge, S.T. (1985) *Biographia Literaria: Biographical Sketches of my Literary Life & Opinions.* New Jersey: Princeton University Press.

Dwarf Fortress Wiki (2007) *Main Page.* Available at: https://dwarffortresswiki.org/index.php/Main_Page (Accessed: 12 October 2020)

Dwarf Fortress Wiki (2007) *DF2014:Creature.* Available at: https://dwarffortresswiki.org/index.php/DF2014:Creature (Accessed: 12 October 2020)

Dwarf Fortress Wiki (2007) *DF2014:Personality Traits.* Available at: https://dwarffortresswiki.org/index.php/DF2014:Personality_trait#Goals (Accessed: 13 October 2020)

Dwarf Fortress Wiki (2007) *DF2014:Memories.* Available at: https://dwarffortresswiki.org/index.php/DF2014:Memories (Accessed: 13 October 2020)

Eccles, V. (2017) *Procedurally Generated Background Characters*. B.A. Dissertation. University of Dublin Trinity College. Available at: https://www.scss.tcd.ie/publications/theses/diss/2017/TCD-SCSS-DISSERTATION-2017-036.pdf (Accessed: 14 November 2020).

Elder Scrolls Wiki (2020) *Alessia Ottus.* Available at: https://elderscrolls.fandom.com/wiki/Alessia_Ottus (Accessed: 20 January 2021)

Elder Scrolls Wiki (2018) *Hastrel Ottus.* Available at: https://elderscrolls.fandom.com/wiki/Hastrel_Ottus (Accessed: 20 January 2021)

Elder Scrolls Wiki (2019) *Ida Ottus.* Available at: https://elderscrolls.fandom.com/wiki/Ida_Ottus (Accessed: 20 January 2021)

Elite Dangerous Wiki (2021) *Elite Dangerous – Galaxy*. Available at: https://elite-dangerous.fandom.com/wiki/Galaxy#:~:text=Elite%20Dangerous%20contains%20a%20vast,modeled%20on%20actual%20galactic%20charts (Accessed: 27 March 2021).

Frontier Developments (2014) *Elite Dangerous* [Video game]. Frontier Developments.

Gomes, P. F., Martinho, A., Paiva A. and Jhala, A. (2013) 'Metrics for Character Believability in Interactive Narrative', *International Conference on Interactive Digital Storytelling,* 8230, DOI:10.1007/978-3-319-02756-2_27.

Hello Games (2016) *No Man's Sky* [Video game]. Hello Games.

Ibarrola-Rivas, M. J., Kastner, T. and Nonhebel, S. (2016) 'How Much Time Does a Farmer Spend to Produce My Food? An International Comparison of the Impact of Diets and Mechanization', *Resources,* 5(4), pp. 47-60, doi: 10.3390/resources5040047

Jensen, K. T. (2020) *Watch Dogs: Legion Might Change the World of NPCs Forever.* Available at: https://uk.pcmag.com/migrated-84555-gaming/129661/watch-dogs-legion-might-change-the-world-of-npcs-forever (Accessed: 27 March 2021)

Llewellyn, M. (2020) *The 10 Smallest Open World Games (Based on the Size of Their Maps).* Available at: https://gamerant.com/yakuza-gta-3-small-world-maps-open-world-games/ (Accessed: 24 March 2021).

Ludion Studios (2018) *Rimworld* [Video game]. Ludion Studios

Maccima Games (2020) *Ruinarch* [Video game]. Squeaky Wheel Studio Inc.

Matulef, J. (2016) *Event[0] is 2001 Meets Firewatch, Due this September*. Available at: https://www.eurogamer.net/articles/2016-07-13-event-0-is-2001-meets-firewatch-due-this-september (Accessed: 27 March 2021).

Maxis (2000) *The Sims* [Video game]. Electronic Arts.

McCrae, R. R. and John, O. P. (1992) 'An Introduction to the Five-Factor Model and its Applications', *Journal of Personality,* 60(2), pp. 175-215.

Mikkelson, N. (2017) *Worldbuilding with NPC Dialogue: A Beginner's Guide.* Available at: https://www.gamasutra.com/blogs/NatalieMikkelson/20171109/309244/Worldbuilding_With_NPC_Dialogue_A_Beginners_Guide.php. (Accessed: 27 March 2021).

Myers & Briggs Foundation (2021 Updated) *MBTI Basics.* Available at: https://www.myersbriggs.org/my-mbti-personality-type/mbti-basics/ (Accessed: 20 October 2020)

National Records of Scotland (2021 Updated) *Names.* Available at: https://www.nrscotland.gov.uk/statistics-and-data/statistics/statistics-by-theme/vital-events/names (Accessed: 20 December 2020)

National Health Service/NHS (2018) *Overview: Menopause.* Available at: https://www.nhs.uk/conditions/menopause/ (Accessed: 12 February 2021)

Ocelot Society (2016) *Event[0]* [Video game]. Ocelot Society.

Rimworld Wiki (2018) *Main Page.* Available at: https://rimworldwiki.com/ (Accessed: 17 October 2020)

Roccas, S., Sagiv, L., Schwartz, S. H. and Knafo, A. (2002) 'The Big Five Personality Factors and Personal Values', *Personality and Social Psychology Bulletin,* 28(6), pp. 789-801.

Ryan, J. (2018) *Curating Simulated Storyworlds.* Doctorate. Dissertation. University of California Santa Cruz. Available at: https://escholarship.org/uc/item/1340j5h2#main (Accessed: 19 October 2020), pp. 371-460.

Saksena, S. (2020) *Watch Dogs: Legion Max Team Size; Get to Know the Max Number of Recruits for Your Team.* Available at: https://www.republicworld.com/technology-news/gaming/watch-dogs-legion-max-team-size-get-to-know-the-max-number-of-recruits-for-your-team.html (Accessed: 27 March 2021).

SNORRI STURLUSON, & ANDERSON, R. B. (1880, *written circa* 1220). *The Younger Edda: also called Snorre's Edda, or the Prose Edda. An*

English version of the foreword ; The fooling of Gylfe, the afterword ; Brage's talk, the afterword to Brage's talk, and the important passages in the Poetical diction (Skáldskaparmál), with an introduction, notes, vocabulary, and index. Chicago, S.C. Griggs and Co.

Steam (2021) *Dwarf Fortress.* Available at: https://store.steampowered.com/app/975370/Dwarf_Fortress/ (Accessed: 24 April 2021)

SteamCharts (2021) *Elder Scrolls IV: Oblivion.* Available at: https://steamcharts.com/app/22330 (Accessed: 24 April 2021)

Ubisoft Montreal (2007) *Assassin's Creed* [Video game]. Ubisoft.

Ubisoft Toronto (2020) *Watch Dogs: Legion* [Video game]. Ubisoft.

UESP (2017) *Elder Scrolls Oblivion Credits.* Available at: https://en.uesp.net/wiki/Oblivion:Credits (Accessed: 26 March 2021).

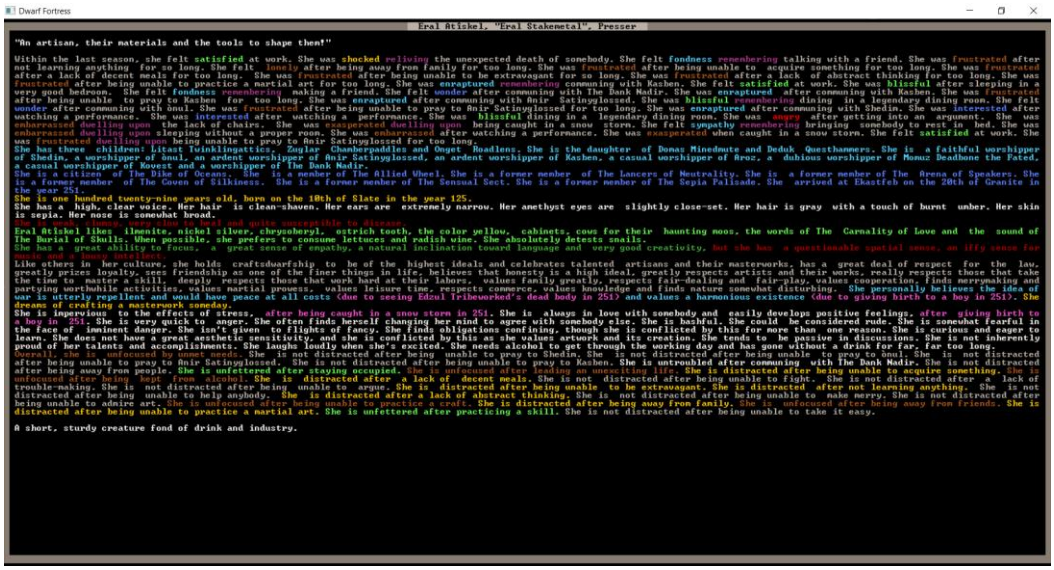
University of Oregon (2002) *Planning Analysis: Calculating Growth Rates.* Available at: https://pages.uoregon.edu/rgp/PPPM613/class8a.htm#:~:text=Calculating%20Growth%20Rates&text=The%20annual%20percentage%20growth%20rate,N%2C%20the%20number%20of%20years.&text=In%201980%2C%20the%20population%20in%20Lane%20County%20was%20250%2C000 (Accessed: 20 February 2021)

Zero Punctuation (2008) *Elder Scrolls IV: Oblivion (Transcript).* Available at: https://zeropunctuation.fandom.com/wiki/The_Elder_Scrolls_IV:_Oblivion#Transcript (Accessed: 25 March 2021).

Appendices

Appendix A – Dwarf Brains

Tekkud Fathomrist, Father



Eral Atiskel, Mother

Onget Roadlens, Child

Appendix B - Questionnaire

NPC Dialogue Review

You will be presented with dialogue from 3 different games. Please rate the overall engagement of the collective dialogue from each source using the scales below them.

Context for the relationships between characters will be provided before the dialogue.

Questions the user can ask will be in parenthesis, and the answers in square brackets as in this example. reactionary dialogue to the user engaging in conversation will be shown in braces before any questions.

NPC1
{Reaction}
  (Question 1)
    [Answer 1]

If the NPC does not have questions to be asked but still generates dialogue, it will be displayed as follows;

NPC2
  [Dialogue 1]
  [Dialogue 2]

Overview of question formatting

Ida Ottus is the daughter of Alessia Ottus (her mother) and Hastrel Ottus (her father). They are commoners.

Ida Ottus
{My name is Ida Ottus, and I'd like to talk about Mother Mara.}
(Imperial City)
[Mara's blessing on you, and on all her children. Speak to her at the Temple of the One. Ask Mother Mara; she will shower your heart with love.]

Alessia Ottus
{My name is Alessia Ottus. And I'd like to talk about the Nine Divines.}
(Imperial City)
[St. Alessia has blessed our city and all its people. Visit the Temple of the One, and renew your faith.]

Hastrel Ottus
{My name is Hastrel Ottus, and I'd like to talk about Akatosh.}
(Imperial City)
[Akatosh speaks to us all, but we never listen. Go to the Temple of the One. Read the Covenants. Praise the Nine!]

Dialogue from Oblivion

Archie Cheung is a Hunter. Clara Cheung is a weaver. They are married with 4 children, the eldest of whom is Maureen, who is a weaver.

Clara Cheung
(What happened to your torso?)
[I scarred my torso when I was 18]
(How old was your mother when she died?)
[My mother is actually still alive]
(Do you have any children?)
[I have 4 wonderful children!]
(How old was your mother when she had you?)
[My mother was 25 years old when she had me]

Archie Cheung
(How old were you when you married Clara?)
[I married Clara when I was 18 and they were 19]
(What happened to your right leg?)
[I crippled my right leg when I was 39]
(What did you do when you turned 16?)
[I became an Hunter once I became an adult.]
(How old was your mother when she died?)
[My mother is actually still alive]

Maureen Mclay
(Do you have any children?)
[I don't have any children]
(How old was your mother when she had you?)
[My mother was 23 years old when she had me]
(Do you have any shared friends with Adele Compton?)
[No. Adele and I don't share any friends]
(Who were your parents?)
[My mother was Clara and my father was Archie]

Dialogue from the generated village

Onget Tuloncerol is a child. Their mother is Eral Atiskel, a presser, and their father is Tekkud Fathomrist, a miner.

Onget Tuloncerol
[He is quite clumsy]
[Onget Tuloncerol likes sphalerite, trifle pewter, wax opal, giant walrus leather, bucklers, dogs for their loyalty, bumblebees for their woolly appearance...]
[He has a great affinity for language and a great feel for the surrounding space]
[He doesn't handle stress well, a turn-around after being caught in a snow storm in 252]
[He finds helping others emotionally rewarding]

Eral Atiskel
[She personally believes the idea of war is utterly repellent and would have peace at all costs]
[She is one hundred and twenty-nine years old, born on the 10th of Slate in the year 125]
[She absolutely detests snails]
[Overall she is unfocussed by unmet needs]
[She has three children: Litast Twinklingattics, Zuglar Chamberpaddles and Onget Roadlens]

Tekkud Fathomrist
[He is shaken after suffering a major injury]
[Within the last season he felt satisfied remembering work]
[He is a worshipper of Shedim]
[He is very thin. His upper body is gone]
[He is very rarely sick, very agile and slow to tire]

Dialogue from Dwarf Fortress

Appendix C – Link to Example Questionnaire

https://forms.gle/BpUPjkkQtXL7tqoU6

This link leads to a copy of the questionnaire where the questions are no longer mandatory to complete to progress.

Appendix D – Results

https://docs.google.com/spreadsheets/d/19yhIfouAxHKayAu2CQyfQjrWjh56yZT2CBa0B3_Y6tY/edit?usp=sharing